2004

Editorial: Rapid system prototyping

Wills, Linda M.; Kordon, Fabrice; Luqi

Elsevier

Wills, Linda M., Fabrice Kordon and Luqi. "Rapid system prototyping." Journal of Systems and Software 70.3 (2004): 225-227.
http://hdl.handle.net/10945/65716

Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®
The Journal of Systems and Software 70 (2004) 225–227
The Journal of Systems and Software
www.elsevier.com/locate/jss

Editorial

Rapid system prototyping

Designers of computer-based systems are faced with difficult technical challenges due to ever increasing complexity and market pressures. The ability to rapidly explore design alternatives and to detect design errors as early as possible is critical in creating high quality products within short development timeframes.

What is prototyping?

Rapid prototyping is an early design exploration and validation process in which executable models of a system are created that reflect some subset of properties of interest. These are evaluated to quickly uncover design errors or undesirable properties that help refine and flesh out requirements. The key idea is to be able to create revelatory prototypes quickly and with less effort than for full system implementation and production. Rapid prototyping techniques have been an integral part of most hardware/software system design methodologies for several years. These include the use of executable specification languages, modeling and simulation environments, automated code generation techniques, test case generation and management, and evolutionary development methodologies.

Interest in the field of rapid system prototyping research has been growing vigorously from both the hardware and software communities, as embedded computer-based systems have become pervasive in the world of technology and everyday life. As these systems become more advanced and sophisticated, new challenges arise including the need for: scalability to large scale systems, distributed communication between interconnected components across heterogeneous and changing (possibly mobile) platforms, readability of executable models for collaboration with the user and for ease of maintenance, support for interdisciplinary, collaborative design, and evolvability and portability of designs as requirements change and technology advances.

The RSP conference series

Every summer since 1990, a group of researchers and practitioners has met to discuss recent innovations, trends, and industrial experiences in this field at the IEEE International Workshop on Rapid System Prototyping (RSP). This special issue is devoted to expanded versions of the best papers from the 12th RSP, which was held in Monterey, CA in June 2001.

The prototyping applications range from FPGA-based embedded hardware to interactive software systems to networked communication systems. Most of the papers presented at this workshop deal with work that is firmly rooted in industrial applications, such as portable multimedia systems, electronic design automation, and distributed, embedded control systems in the automotive industry. Many papers describe collaborative efforts between researchers from industry and universities in both the software and hardware communities.

Selected papers for JSS

There were 57 papers submitted to RSP 2001 out of which 30 were selected. The program committee recommended a small subset of these as candidates for the special issue, based on their innovative contributions and practical implications. Of this subset, six were selected after a two-phase review process by experts in rapid prototyping and by experts in the application domains targeted by the papers. This issue highlights a representative cross-section of the work that is going on in this area.

One of the trends in rapid prototyping is to reduce design complexity by decoupling component communication from specifications of the components' behavior. This is particularly evident in the papers by Cesario et al. and by Guisto et al. Both papers seek to reduce development and production costs and shorten design time through this separation of concerns.

Cesario et al. present a hierarchical, object-oriented model (Colif) focused on modeling interconnections between hardware and software components in multiprocessor system-on-a-chip (MPSOC) architectures. Its primary contribution is that it attacks the important and difficult problem of capturing and *automatically generating* communication interfaces between hardware and software architectural components during prototyping.

0164-1212/$ - see front matter
doi:10.1016/S0164-1212(03)00070-0

In Colif, communication interfaces separate hardware processors from the communication interconnect IP, freeing the processors from communication management. Likewise, software tasks are isolated from the hardware and interconnection network through OS-based wrapping. Colif design tools support automatic generation of hardware/software interfaces and of custom embedded operating systems. This facilitates rapid design exploration since many different alternative communication schemes (topologies/protocols) and different task scheduling and resource management policies can be easily captured, experimented with, evaluated, and refined.

Like Colif, the paper by Giusto and Demmeler presents a methodology that decouples behavioral modeling from communication implementation considerations. However, while Cesario et al. focus on interface generation, Giusto and Demmeler focus on automatically configuring communication protocols during design refinement. In particular, the behavioral model of the functional network of the distributed control algorithm is captured with zero-time execution and communications assumptions. These idealized models are then automatically annotated with performance models that include realistic execution latencies and communication delays, so that the designer can assess whether the chosen hardware/software architecture meets real-time environmental constraints. The design refinement step refines the abstract communication models in terms of real communication protocols and it maps the abstract behaviors to actual HW/SW implementations. This shortens design time by automating a step that is normally performed manually (at great cost) and it allows adaptability in changing the distributed architecture and protocol configuration independent of the functional network/behavioral model.

These papers (and indeed most of the papers in this issue) describe experiences with industrial-strength case studies. Colif is validated in the context of a multiprocessor system-on-a-chip design of a VDSL modem. Giusto and Demmeler focuses on prototyping distributed architectures of networked electronic control units, primarily targeting automotive applications. The third paper in this special issue (by Deppe et al.) also targets complex distributed control systems, such as those in automobiles and autonomous vehicles, consisting of complex networks of interconnected control units. The authors describe a case study of using a layered design approach for complex control systems, focusing on complicated mechatronic applications in the domain of autonomous vehicle control.

These systems typically consist of mechanical, hydraulic, and electrical components in addition to computer hardware and software components. Prototyping in these complex, real-world applications requires interdisciplinary knowledge and collaboration. The primary innovation of the paper by Deppe et al. is that they provide a coherent methodology for supporting this interdisciplinary work. They present an integrative software environment (CAMeL) for co-operative design, simulation, and optimization. This supports multi-domain modeling (mechanics, hydraulics, information processing) at high levels of abstraction (e.g., topological and mathematical levels).

The distributed nature of many of these applications adds to development complexity, cost, and vulnerability to errors. Automatically verifiable models and efficient, accurate code generation from these models are of tremendous benefit to embedded systems developers. The paper by Hansel et al. describes an automated code generation technique that produces distributed C++ prototypes from high level system models written in VPL. These models capture specifications of hierarchical systems of concurrent processes with synchronous message passing interaction semantics. The approach targets a variety of execution platforms and RTOSs by building on the widely used adaptive communication environment (ACE) library for synchronous communication. This library encapsulates low-level interprocess communication issues, which enables portability and readability. This work highlights a trend in system design in which component-based software design tools are integrated with domain-specific formal specification and "lightweight" verification techniques.

Another challenge for computer-based system development is flexibly accommodating changes in technology (e.g., FPGA advances) and in requirements, making them reusable as well as portable. Ernst et al. address this challenge in the prototyping of public key cryptosystems. Their paper targets the important application of securing confidential data transfers (e.g., for internet banking) on public communication networks by improving the digital signature verification rate. The paper focuses on mapping the most time consuming part of cryptography algorithms to FPGA-based reconfigurable hardware implementations. The resulting hybrid hardware/software cryptosystem exhibits high performance, but more importantly portability and evolvability. These are critical to flexibly making design tradeoffs with respect to security-level performance. This approach supports rapid design exploration using a novel model generator, which produces VHDL models that feed into commercially available synthesis tools. The approach is flexible with respect to the key size (which varies with security-level requirements) as well as accommodating rapid advances in FPGA technology.

Finally, extreme programming, which advocates incremental, evolutionary software development, is an increasingly popular form of rapid prototyping. The executable artifacts that are produced on each development cycle serve as prototypes that must be evaluated in guiding the development process. The paper by Prets-

chner et al. addresses the problem of generating test sequences for these critical evaluation steps in developing reactive systems. A key insight is that useful test cases can be derived automatically from graphical specification models as well as from the source code implementations. The authors formulate test case generation as a search problem using constraint logic programming. The paper describes the authors' experience in using this technique with a large-scale industrial application.

Future of RSP

Rapid system prototyping is now widely accepted in system development from hardware to software. It integrates various tools, including modelers, simulators, model checkers, and code generators, and methodologies with the knowledge of highly skilled people in technical domains (e.g., software/hardware partitioning for system-on-a-chip designs, management of parallelism in distributed systems, etc.). The progress in this field is targeting a critical need in industry for expertise in these domains. This special issue provides a snapshot of the current trends and challenges in this rapidly growing field. The papers featured here provide a richly detailed perspective on the state of the field of rapid system prototyping.

Acknowledgements

The guest editors thank the reviewers of the papers for this special issue for their feedback on multiple versions of the papers and to the RSP 2001 program committee for their recommendations. They are also grateful to Robert Glass (Editor Emeritus of JSS) and the JSS staff for their help in producing this special issue.

Linda M. Wills
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0250, USA
E-mail address: linda.wills@ece.gatech.edu

Fabrice Kordon
Computer Science Department
Université Pierre & Marie
Curie 4 place Jussieu
75252 Paris Cedex 05, France
E-mail address: fabrice.kordon@lip6.fr

Luqi
Computer Science Department
Naval Postgraduate School
1 University Cir.
Monterey, CA 93943-5001, USA
E-mail address: luqi@cs.nps.navy.mil